CSc 110, Autumn 2016

Lecture 9: `input; if/else`

Adapted from slides by Marty Stepp and Stuart Reges

Interactive programs

interactive program: Reads input from the console.

- While the program runs, it asks the user to type input.
- The input typed by the user is stored in variables in the code.

- Can be tricky; users are unpredictable and misbehave.
- But interactive programs have more interesting behavior.

`input`

- **`input`**: An function that can read input from the user.

- Using an `input` object to read console input:

  name = `input(`**prompt**`)`

  - Example:

  ```
  name = input("type your name: ")
  ```

    - The variable `name` will store the value the user typed in

input example

```
def main():
    age = input("How old are you? ")

    years = 65 - age
    print(years + " years until retirement!")
```

age 29

- Console (user input underlined):

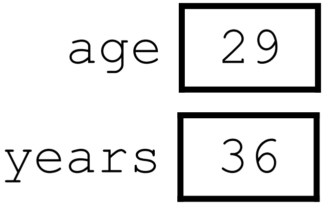
```
How old are you? 29
```

```
Traceback (most recent call last):
  File "<pyshell#13>", line 1, in <module>
    print(65 - age)
TypeError: unsupported operand type(s) for -:
'int' and 'str'
```

input example

```
def main():
    age = int(input("How old are you? "))

    years = 65 - age
    print(str(years) + " years until retirement!")
```



age 29

years 36

- Console (user input underlined):

```
How old are you? 29
36 years until retirement!
```

The `if/else` statement

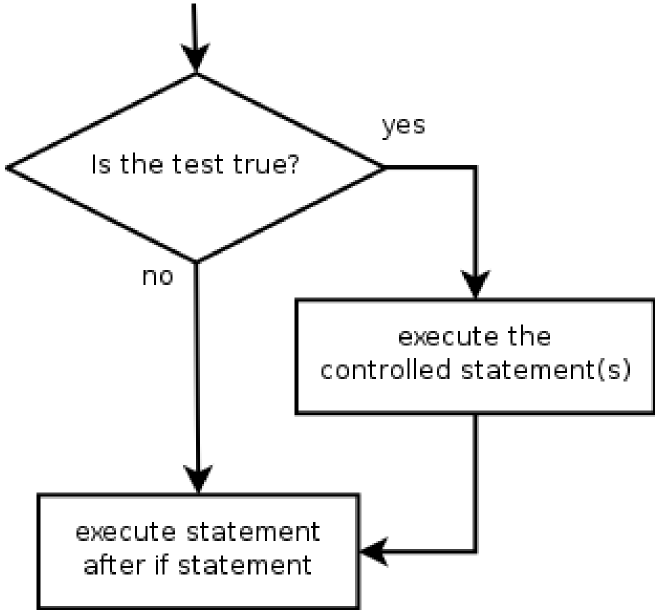
The `if` statement

Executes a block of statements only if a test is true

```
if (test):
    statement
    ...
    statement
```

- Example:

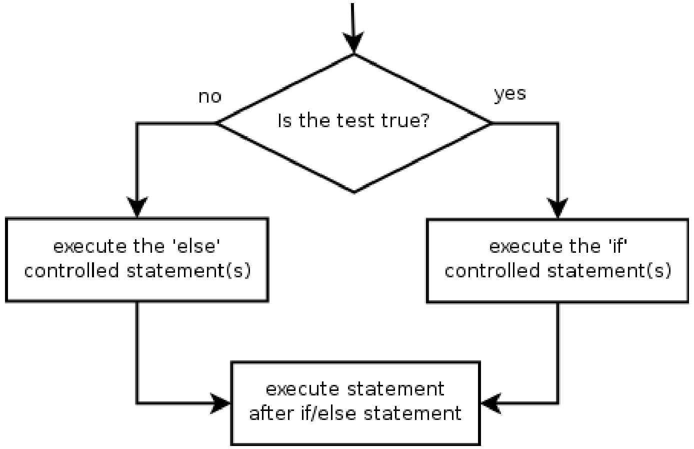
```
gpa = float(input("gpa? "))
if (gpa >= 2.0):
    print("Application accepted.")
```


Is the test true?
yes
no
execute the controlled statement(s)
execute statement after if statement

The `if/else` statement

Executes one block if a test is true, another if false

```
if (test):
    statement(s)
else:
    statement(s)
```


Is the test true?
no
yes
execute the 'else' controlled statement(s)
execute the 'if' controlled statement(s)
execute statement after if/else statement

- Example:

```
gpa = float(input("gpa? "))
if (gpa >= 2.0):
    print("Welcome to Mars University!")
else:
    print("Application denied.")
```

Relational expressions

- `if` statements use logical tests.

```
if (i <= 10): ...
```

  - These are `boolean` expressions

- Tests use *relational operators*:

Operator	Meaning	Example	Value
`==`	equals	`1 + 1 == 2`	`True`

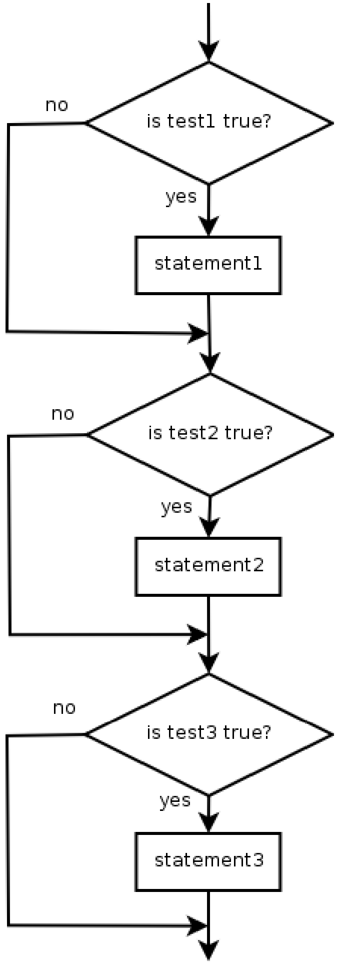
`!=`
`<>`
	does not equal	`3.2 != 2.5`
		`3.2 <> 2.5`
			`True`

`<`	less than	`10 < 5`	`False`
`>`	greater than	`10 > 5`	`True`
`<=`	less than or equal to	`126 <= 100`	`False`
`>=`	greater than or equal to	`5.0 >= 5.0`	`True`

Misuse of `if`

- What's wrong with the following code?

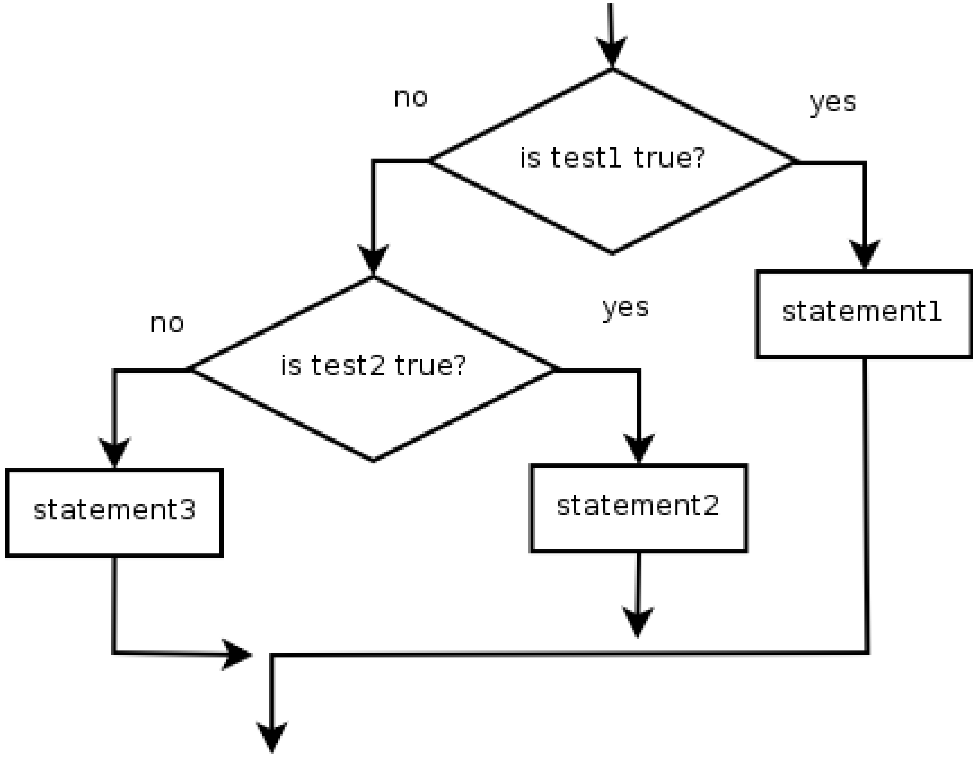
```
percent = float(input("What percentage did you earn? "))
if (percent >= 90):
    print("You got an A!")
if (percent >= 80):
    print("You got a B!")
if (percent >= 70):
    print("You got a C!")
if (percent >= 60):
    print("You got a D!")
if (percent < 60):
    print("You got an F!")
...
```



Nested `if/else`

Chooses between outcomes using many tests

```
if (test):
      statement(s)
elif (test):
      statement(s)
else:
      statement(s)
```



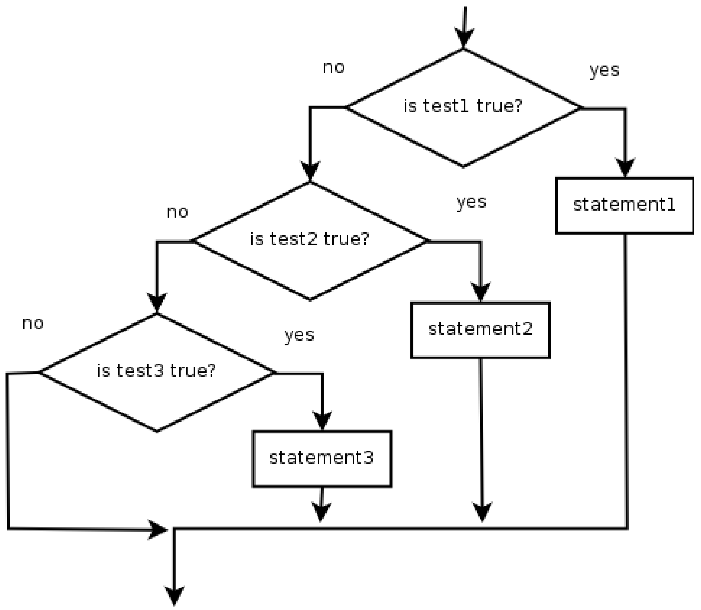
- Example:

```
if (x > 0):
    print("Positive")
elif (x < 0):
    print("Negative")
else:
    print("Zero")
```

Nested `if/else/if`

- If it ends with `else`, exactly one path must be taken.
- If it ends with `if`, the code might not execute any path.

```
if (test):
    statement(s)
elif (test):
    statement(s)
elif (test):
    statement(s)
```



- Example:

```
if (place == 1):
    print("Gold medal!")
elif (place == 2):
    print("Silver medal!")
elif (place == 3):
    print("Bronze medal.")
```

Nested `if` structures

- exactly 1 path *(mutually exclusive)*

```
if (test):
    statement(s)
elif (test):
    statement(s)
else:
    statement(s)
```

- 0 or 1 path *(mutually exclusive)*

```
if (test):
    statement(s)
elif (test):
    statement(s)
elif (test):
    statement(s)
```

- 0, 1, or many paths *(independent tests; not exclusive)*

```
if (test):
    statement(s)

if (test):
    statement(s)

if (test):
    statement(s)
```

Which nested `if/else`?

- **(1) if/if/if (2) nested if/else (3) nested if/else/if**
  - Whether a user is lower, middle, or upper-class based on income.
    - **(2)** nested `if / else if / else`
  - Whether you made the dean's list (GPA ≥ 3.8) or honor roll (3.5-3.8).
    - **(3)** nested `if / else if`
  - Whether a number is divisible by 2, 3, and/or 5.
    - **(1)** sequential `if / if / if`
  - Computing a grade of A, B, C, D, or F based on a percentage.
    - **(2)** nested `if / else if / else if / else if / else`